Flyer, N., Fornberg, B., Bayona, V., Barnett, G. A. (2016). On the role of polynomials in RBF-FD approximations: I. Interpolation and accuracy. *Journal of Computational Physics*, 321, 21-38.

DOI: https://doi.org/10.1016/j.jcp.2016.05.026

On the role of polynomials in RBF-FD approximations: I. Interpolation and accuracy

Natasha Flyer *
Institute for Mathematics Applied to Geosciences
National Center for Atmospheric Research
Boulder, CO 80305, USA

Bengt Fornberg †
Department of Applied Mathematics
University of Colorado
Boulder, CO 80309, USA

Victor Bayona ‡
Department of Mathematical Sciences
University of Bath
Bath, BA2 7AY, UK

Gregory A. Barnett §
Department of Applied Mathematics
University of Colorado
Boulder, CO 80309, USA

May 3, 2016

Abstract

Radial basis function-generated finite difference (RBF-FD) approximations generalize classical grid-based finite differences (FD) from lattice-based to scattered node layouts. This greatly increases the geometric flexibility of the discretizations and makes it easier to carry out local refinement in critical areas. Many different types of radial functions have been considered in this RBF-FD context. In this study, we find that (i) polyharmonic splines (PHS) in conjunction with supplementary polynomials provide a very simple way to defeat stagnation (also known as saturation) error and (ii) give particularly good accuracy for the tasks of interpolation and derivative approximations without the hassle of determining a shape parameter. In follow-up studies, we will focus on how to best use these hybrid RBF polynomial bases for FD approximations in the contexts of solving elliptic and hyperbolic type PDEs.

* *Email*: flyer@ucar.edu
† *Email*: fornberg@colorado.edu
‡ *Email*: victor.bayona.revilla@gmail.com
§ *Email*: gregory.barnett@colorado.edu

1 Introduction

With RBFs increasingly being used for large-scale applications, global RBFs are often being replaced by local RBF-FD approximations [1, 2, 5, 6, 13, 17, 27, 29, 30, 32, 34]. This is to be expected, for several reasons: (i) global RBFs lead to full matrices and thus can be computationally prohibitive for large-scale problems while RBF-FD lead to extremely sparse matrices and thus are computationally cheap, (ii) interpolation and derivatives are local properties of a function, so local approximations are natural in the context, and (iii) multi-core and distributed memory computing greatly benefits from the spatial locality RBF-FD approximations offer [3, 9, 33]. In the fore-mentioned studies, Gaussian (GA) or Multiquadric (MQ) RBFs are used for the approximations and will in all cases lead to stagnation (saturation) error under refinement (unless a stable algorithm such as RBF-GA is used [20]).

This paper was written in tandem with [12], introducing a novel concept for controlling stagnation errors through the use of high-order polynomials together with RBFs. For the current study, the first in a sequence of four, the analysis is done in the context of interpolation for both GA and polyharmonic splines (PHS). For PHS, the control of stagnation errors when evaluating derivatives given known function values is also considered. It is shown that under refinement, the polynomial degree, not the RBF, controls the rate of convergence, making PHS very attractive to work with since there is then no need to consider a shape parameter. However, it should be noted that the RBFs help to reduce approximation errors in comparison to using classical polynomial least-squares by enhancing the approximation space.

We first briefly review global RBFs in Section 2 and discuss historically why polynomial terms were added. Then in Section 3, we turn to how analyzing polynomial reproduction gives insights into stagnation errors for global RBFs, which will be seen to be the same mechanism for RBF-FD in the case of GA. Section 4 gives a brief overview of how to calculate the RBF-FD weights with polynomial augmentation. The main novelties of this study are contained in Section 5, where we discuss the ramifications of including high-order polynomials when creating RBF-FD approximations. Key issues concern the effect on rate of convergence, accuracy and conditioning. RBF-FD approximations based on PHS type turn out to be of particular interest. Lastly, we compare this novel approach against using polynomial least-squares approximations, a common approximation method in the scientific community, and show the superiority of using an augmented polynomial-RBF basis. The main conclusions are summarized in Section 7.

2 Some basic RBF relations

A RBF interpolant $s(\underline{x})$ to scattered data in d dimensions (d-D) takes the form

$$s(\underline{x}) = \sum_{k=1}^{n} \lambda_k \phi(||\underline{x} - \underline{x}_k||) \,, \tag{1}$$

where $||\cdot||$ denotes the standard Euclidean norm. Table 1 lists a number of common choices for the *radial function* $\phi(r = ||\cdot||)$.

With data f_k at node $\underline{x}_k$, $k = 1, 2, \ldots, n$, the coefficients λ_k in (1) can be found by solving the

Type of basis function	**Radial function** $\phi(r)$	**Reproduced degree**
BE (Bessel; $p = 1, 2, \ldots, d$)	$J_{p/2-1}(\varepsilon r)/(\varepsilon r)^{p/2-1}$	all
Polyharmonic Spline(PHS)	r^{2m-1} or $r^{2m}\log r$ $m \in \mathbb{N}$	$d + (m-1)$
Ex. of PHS:		
Cubic	r^3	$d+2$
TPS (Thin Plate Spline)	$r^2 \log r$	$d+1$
Linear	r	d
MQ (Multiquadric)	$\sqrt{1+(\varepsilon r)^2}$	d
IMQ (Inverse MQ)	$1/\sqrt{1+(\varepsilon r)^2}$	$d-2$
IQ (Inverse quadratic)	$1/(1+(\varepsilon r)^2)$	$d-3$
GA (Gaussian)	$e^{-(\varepsilon r)^2}$	none
RBFs with compact support	various formulas	none

Table 1: Common RBF types, and the degrees of polynomials global RBFs based on these can reproduce exactly on general infinite d-dimensional (d-D) lattices. The infinitely differentiable RBFs include a shape parameter $\varepsilon > 0$ and

linear system

$$\begin{bmatrix} \phi(||\underline{x}_1 - \underline{x}_1||) & \phi(||\underline{x}_1 - \underline{x}_2||) & \cdots & \phi(||\underline{x}_1 - \underline{x}_n||) \\ \phi(||\underline{x}_2 - \underline{x}_1||) & \phi(||\underline{x}_2 - \underline{x}_2||) & \cdots & \phi(||\underline{x}_2 - \underline{x}_n||) \\ \vdots & \vdots & & \vdots \\ \phi(||\underline{x}_n - \underline{x}_1||) & \phi(||\underline{x}_n - \underline{x}_2||) & \cdots & \phi(||\underline{x}_n - \underline{x}_n||) \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix}. \tag{2}$$

Remarkably, the symmetric coefficient matrix A is guaranteed to be non-singular for many choices of $\phi(r)$, no matter how the n nodes (assumed distinct) are scattered in any number dimensions.

2.1 Global RBFs with polynomials

Equation (1) is often modified to also include polynomial terms for conditionally positive definite radial functions (e.g. PHS, MQ), together with matching constraints on the expansion coefficients. For example, including up to linear terms in 2-D would amount to

$$\begin{aligned} s(\underline{x}) &= \sum_{k=1}^{n} \lambda_k \phi(||\underline{x} - \underline{x}_k||) \; + \; \gamma_1 + (\gamma_2 x + \gamma_3 y) \; , \\ \text{with the constraints} \qquad & \textstyle\sum_{k=1}^{n} \lambda_k = \sum_{k=1}^{n} \lambda_k x_k = \sum_{k=1}^{n} \lambda_k y_k = 0, \end{aligned} \tag{3}$$

given the notation $\underline{x} = (x, y)$ and $\underline{x}_k = (x_k, y_k)$. The counterpart to (1) will then be

$$\left[\begin{array}{ccc|ccc} & & & 1 & x_1 & y_1 \\ & A & & \vdots & \vdots & \vdots \\ & & & 1 & x_n & y_n \\ \hline 1 & \cdots & 1 & 0 & 0 & 0 \\ x_1 & \cdots & x_n & 0 & 0 & 0 \\ y_1 & \cdots & y_n & 0 & 0 & 0 \end{array}\right] \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_n \\ \gamma_1 \\ \gamma_2 \\ \gamma_3 \end{bmatrix} = \begin{bmatrix} f_1 \\ \vdots \\ f_n \\ 0 \\ 0 \\ 0 \end{bmatrix}, \tag{4}$$

where A is the same matrix as in (1). Historically, the main reasons for including polynomial terms and their constraints for radial functions such as PHS and MQ are

1. To ensure that the linear system (2) is uniquely solvable when using conditionally positive (or negative) definite radial functions (assuming the node set is unisolvent on the space spanned by the polynomial terms), see Chapter 7 in [10]. In the constrained parameter space ($\gamma_1, \gamma_2, \gamma_3$ are Lagrange multipliers that constrain the RBF coefficients to the space $P^T\lambda = 0$, where P^T is the lower left block matrix in (4)), the corresponding linear system (4) will then represent positive (or negative) definite operators and thus guarantee a unique solution.

2. Practical experience shows that already including a constant tends to improve the accuracy of derivative approximations, in particular avoiding oscillatory representations of constant data.

3. Including low-order polynomials can improve the accuracy of RBF approximations at domain boundaries as it regularizes the far-field growth of the RBF approximation, see Section 6.1 in [14].

Extensive theory on the solvability of these types of approximations can be found in several monographs [4, 10, 35].

3 The relation between stagnation errors on infinite lattices and RBF-FD stencils for GA RBFs

For global GA RBFs on equispaced infinite lattices, examining how constant data is reproduced adds insights to the RBF-FD case on the issue of *stagnation errors*, defined as errors failing to decrease to zero under continuing node refinement. The mechanism by which stagnation error occurs is well known. By reducing h (with h being a 'typical' node spacing) and, at the same time, increasing ε to keep εh constant leaves the matrix in (2) unchanged and therefore the condition number unaffected. Following this somewhat common practice can however give rise to *stagnation errors* (sometimes denoted saturation errors) - failure of convergence in the $h \to 0$ limit. However, this does not describe what the character of the error is like, as its behavior and distribution is quite different from that of PHS.

Since in the $h \to 0$ limit any smooth function will locally (on the scale of the node spacing) appear increasingly like a constant, it suffices to consider constant data.

First, focusing on 1-D, let the node points be $x_i = i$, $i = -N, \ldots, -1, 0, 1, \ldots, N$. For each N, and with all data values $f_i = 1$, it will hold that $1 = \sum_{k=-N}^{N} \lambda_k e^{-(\varepsilon\,(k-i))^2}$. In the limit of $N \to \infty$, the unique solution will have all the λ_k the same [15]; $\lambda_k = \lambda$ with

$$\lambda = 1/\sum_{k=-\infty}^{\infty} e^{-(\varepsilon\,k)^2}. \tag{5}$$

The GA interpolant then becomes $s(x) = \lambda \sum_{k=-\infty}^{\infty} e^{-(\varepsilon\,(k-x))^2}$. This function $s(x)$ has period 1, but is not identical to 1. Constant data is not identically reproduced.

Extending to an equi-spaced 2-D infinite lattice, a GA interpolant to data identically one will develop a 2-D periodic structure, as seen over one 'period box' in Figure 1 (in the case of $\varepsilon =$

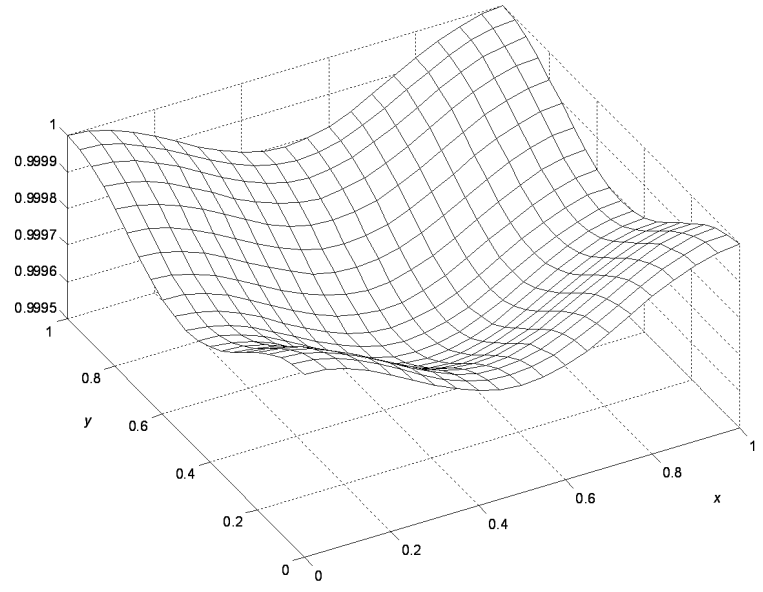

Figure 1: The GA interpolant to node data that is identically one on an infinite 2-D unit spaced lattice, shown in the case of $\varepsilon = 1/h$, with $h = 1$, over its period $[0, 1] \times [0, 1]$.

$1/h$). The deviation from being identically one may seem small (here about $4 \cdot 10^{-4}$), but remains unchanged at this level as $h \to 0$.

Now consider the GA interpolant on $n = 20$ nodes equispaced over $[-1, 1]$ for data that is all equal to one, as plotted in Figure 2. For larger ε, here represented by $\varepsilon = 11$, the interpolant features uniform one-sided oscillations throughout the central part of the interval, just as described by the infinite interval case. When ε is smaller, here $\epsilon = 6$, the error has a quite different character and oscillates both up and down. Right in the transition between the two cases, here $\varepsilon = 9$, there is superb accuracy throughout a quite wide central section. This transition between the two oscillation types is favorable in RBF-FD contexts, corresponding to the very low error regime that is seen near the bottom in the top row of subplots in Figure 6 and will be discussed in that section. This low error regime for intermediate epsilon values was previously noted in a somewhat different context in [22].

Thus as $h \to 0$ and maintaining εh fixed, the character of the error is identical in the case of GA RBFs whether one is considering infinite lattices or a finite stencil. This behavior of GA and does not hold for many other infinitely smooth RBFs or PHS. [1]

4 RBF-FD approximations and calculation of weights

Conceptually, RBF-FD can be seen as an extreme case of overlapping domain decomposition, with a separate domain surrounding each node point, composed of its $n - 1$ nearest neighbors. Thus, unlike lattice-based methods such as classical FD, the stencil shape and thus weights change from node point to node point as is illustrated in Figure 3.

[1] Although Bessel RBFs are a limiting case of GA, having some unique advantageous features [11, 19, 24, 31], they behave quite differently, primarily due to their Fourier transform having compact support and thus deserve a separate analysis that will not be pursued here.

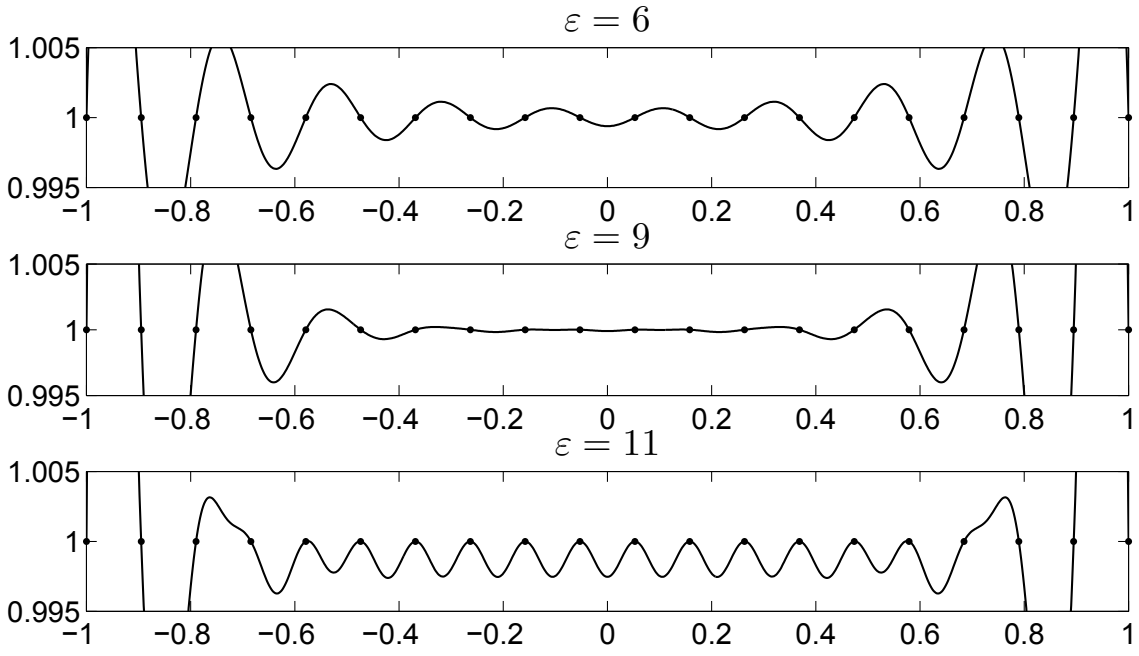

Figure 2: The GA RBF interpolant for data equal to one on $x = [-1, 1]$ using $n = 20$ equi-spaced nodes. The top graph displays the interpolant for $\varepsilon = 6$, the middle $\varepsilon = 9$ and the bottom $\varepsilon = 11$.

Let us assume we want to approximate a linear operator L at a point $\underline{x}_c$ in the domain by a linear combination of the function values, $\{u_k\}_{k=1}^n$, at the nearest n node locations, $\{\underline{x}_k\}_{k=1}^n$; this group of nodes compose the stencil. In other words, we seek weights such that

$$\sum_{k=1}^{n} w_k u_k = (Lu)|_{\underline{x}=\underline{x}_c}, \tag{6}$$

with n and w_k being known as the stencil size and differentiation weights, respectively. To calculate w_k, we enforce that this linear combination of function values be exact for the interpolant $s(\underline{x})$ of the form

$$s(\underline{x}) = \sum_{k=1}^{n} \lambda_k \phi(\|\underline{x} - \underline{x}_k\|) + \sum_{k=1}^{\binom{l+d}{l}} \gamma_k p_k(\underline{x}) \tag{7}$$

with the constraints

$$\sum_{k=1}^{n} \lambda_k p_j(\underline{x}_k) = 0 \quad j = 0, 1, 2, ..., \binom{l+d}{l}$$

, where $p_l(\underline{x})$ are all multivariate polynomials up to degree l in the dimension of the problem d and $\phi(\|\underline{x} - \underline{x}_k\|)$ is an RBF centered at the node $\underline{x}_k$.

As an example, using RBFs and polynomials up to degree $l = 1$ in 2D, the weights are obtained by

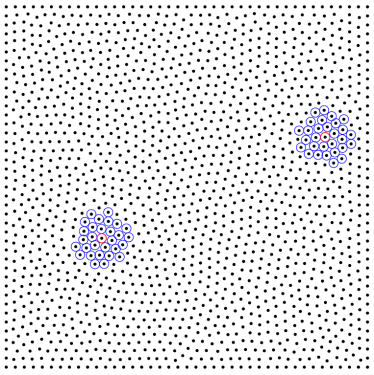

Figure 3: Illustration of two different RBF-FD stencils in a square domain

solving the following linear system

$$\left[\begin{array}{ccc|c|ccc} & & & | & 1 & x_1 & y_1 \\ & A & & | & \vdots & \vdots & \vdots \\ & & & | & 1 & x_n & y_n \\ \hline - & - & - & + & - & - & - \\ \hline 1 & \cdots & 1 & | & & & \\ x_1 & \cdots & x_n & | & & 0 & \\ y_1 & \cdots & y_n & | & & & \end{array}\right] \left[\begin{array}{c} w_1 \\ \vdots \\ w_n \\ \hline - \\ \hline w_{n+1} \\ w_{n+1} \\ w_{n+3} \end{array}\right] = \left[\begin{array}{c} L\phi(||\underline{x}-\underline{x}_1||)|_{\underline{x}=\underline{x}_c} \\ \vdots \\ L\phi(||\underline{x}-\underline{x}_n||)|_{\underline{x}=\underline{x}_c} \\ \hline - \\ \hline L\,1\,|_{\underline{x}=\underline{x}_c} \\ L\,x\,|_{\underline{x}=\underline{x}_c} \\ L\,y\,|_{\underline{x}=\underline{x}_c} \end{array}\right], \tag{8}$$

where the matrix is the same as in (4). Hence, previously quoted non-singularity theorems again apply (see [4, 10, 35]). For interpolation (which can be seen as approximating the zeroth derivative), L would be the identity operator. In the solution vector, the entries w_{n+1}, w_{n+2}, w_{n+3} should be discarded. The pattern of entries in (8) generalizes immediately to more space dimensions and to higher degree polynomials. Note that for coding purposes all node points $\{x, y\}_{k=1}^{n}$ are translated so the evaluation point, $\mathbf{x_c}$, so that it becomes the origin $(0, 0)$. Then, when L operates on the polynomials on the right hand side of (8), all are zero except for the first on $L1$ which becomes one. The derivation of system (8) is given in Section 5.1.4 of [16]. A MATLAB code for calculating RBF-FD weights, $\frac{\partial}{\partial x}$, $\frac{\partial}{\partial y}$ and the 2D Laplacian, using PHS with polynomials is given in Appendix B of [12].

4.1 Least squares interpretation of RBF-FD weights with polynomial augmentation

Analogous to how (4) can be viewed as a constrained optimization problem (as described in point one in Section 2.1), calculating RBF-FD weights with polynomial augmentation can be viewed as the solution of the equality-constrained quadratic programming problem

$$\min_{\boldsymbol{w}} \; J(\boldsymbol{w}) = \frac{1}{2}\boldsymbol{w}^T A \boldsymbol{w} - \boldsymbol{w}^T L\boldsymbol{\phi} \quad \text{subject to} \quad P^T\boldsymbol{w} = L\boldsymbol{p}, \tag{9}$$

where A is the same matrix as in (2), $\boldsymbol{w}$ and $L\boldsymbol{\phi}$ are column vectors formed by w_i and $L\phi(||\underline{x}-\underline{x}_i||)|_{\underline{x}=\underline{x}_c}$, $i = 1, \ldots, n$, respectively. P^T is a $m \times n$ matrix formed by all the $m = \binom{k+d}{k}$ multivariate polynomial terms of total degree less than or equal to k in d variables (notice that $n > m$ for the unisolvency condition to be met), and $L\boldsymbol{p}$ is a $m \times 1$ vector formed by $L\,1\,|_{\underline{x}=\underline{x}_c}$, $L\,x\,|_{\underline{x}=\underline{x}_c}$, $L\,y\,|_{\underline{x}=\underline{x}_c}$, ...

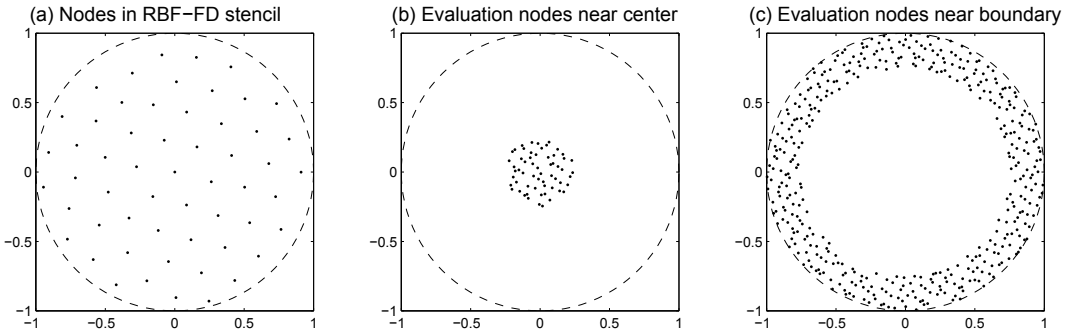

Figure 4: (a) RBF-FD stencil ($n = 56$ nodes) in the case of $R = 1$. Parts (b) and (c): Two sets of evaluation points at which the approximation's interpolation accuracy is evaluated.

This statement follows from using Lagrange multipliers to find the minimum with respect to $\boldsymbol{w}$ and $\boldsymbol{\gamma}$ of the Lagrangian

$$\mathcal{L}(\boldsymbol{w}, \boldsymbol{\gamma}) = \frac{1}{2}\boldsymbol{w}^T A \boldsymbol{w} - \boldsymbol{w}^T L \boldsymbol{\phi} + \boldsymbol{\gamma}^T \left(P^T \boldsymbol{w} - L \boldsymbol{p} \right). \tag{10}$$

For positive definite RBFs, the A matrix is positive definite by definition. For strictly conditionally positive definite RBFs, such as PHS (which is emphasized in this work),

$$\boldsymbol{c}^T A \, \boldsymbol{c} > 0, \quad \forall \boldsymbol{c} \neq \boldsymbol{0} \in \mathbb{R}^n \quad \text{satisfying} \quad P^T \boldsymbol{c} = 0, \tag{11}$$

meaning that A is positive definite for any non-zero vector $\boldsymbol{c}$ in the left null space of P. Therefore, the Lagrangian (10) is convex and has a unique minimum (see Lemma 16.1 in p. 452 in [26]). It follows that $\nabla_{\boldsymbol{w},\boldsymbol{\gamma}} \mathcal{L}(\boldsymbol{w}, \boldsymbol{\gamma}) = 0$ yields the system of equations (8).

Consequently, RBF-FD weights with polynomial augmentation (8) are the solution of the underdetermined linear system $P^T \boldsymbol{w} = L \boldsymbol{p}$ minimizing the quadratic functional in (9). Although we do not use this least squares method to find the RBF-FD weights (but use the system given in (8)), this intriguing observation will hopefully provide an improved understanding of the methodology in the follow-up papers to this study.

5 Exploring interpolation and accuracy for RBF-FD approximations with polynomials

For the present set of test cases, a stencil with $n = 56$ minimum energy-like nodes is considered, as shown within the unit circle ($R = 1$) in Figure 4 (a). In most RBF-FD cases, one is only interested in the accuracy close to the stencil center, although nearly one-sided approximations may become needed in connection with boundaries and interfaces. Comparing interpolation errors at the two sets of evaluation nodes seen in Figures 4 (b,c), it transpires that when the evaluation nodes are near the edge of the stencil the errors were about 100 times larger but otherwise followed equivalent patterns. It therefore suffices to consider the near-center case shown in Figure 4 (b).

We choose as a test function

$$f(x, y) = 1 + \sin(4x) + \cos(3x) + \sin(2y), \tag{12}$$

shown over $R = 1$ in Figure 5 (a). In two steps of refinement, we reduce R from 1 to 0.3 and then to 0.1. The displays in Figure 4 remain unchanged apart from being shrunk in proportion to R.

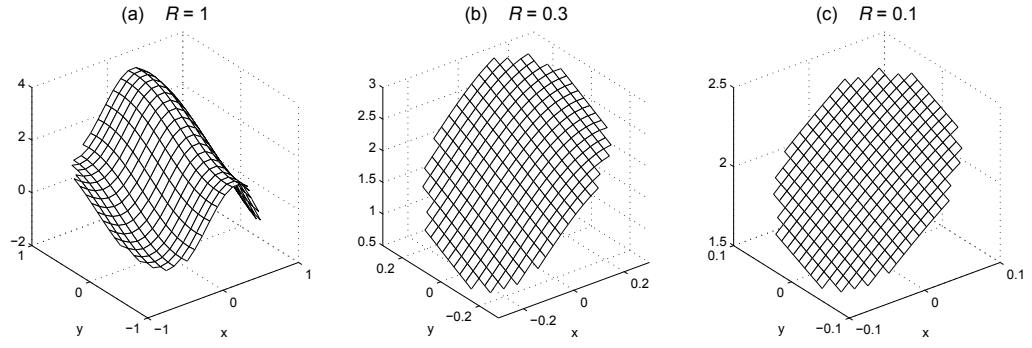

Figure 5: Test function (12), when displayed over $x^2 + y^2 \leq R^2$, with $R = 1$, $R = 0.3$, and $R = 0.1$, respectively.

When zooming in, subplots (b) and (c) of Figure 5 show how the test function appears gradually more as a tilted plane. If the vertical axis scale is fixed, the surfaces would instead increasingly look like a flat plane, i.e. a constant.

All the error results given in Sections 5.1 and 5.2 represent the worst case encountered at any of the evaluation nodes indicated in Figure 4 (b), as mentioned above the most representative case for typical RBF-FD usage. The degree of polynomial terms included is varied from -1 (meaning no polynomials) up through 9. In the present 2-D case with $n = 56$ nodes, the matrix for calculating RBD-FD weights (as illustrated in (8) when appending linear polynomials) becomes singular for degrees 10 and above, as there are 55 polynomials of degree 9, but 66 of degree 10.

5.1 RBF-FD using GA with polynomials

This section describes the test results shown for GA in Figure 6. Several observations can be made:

1. The top row of the figure gives accuracy plots that are calculated in sufficiently high extended precision (80 decimal digits) such that the conditioning of the matrix in (8) does not affect the results. When looking along the left edge of these subplots (corresponding to no supporting polynomials), the error decreases sharply from very large values of ε to small values, reaching a minimum around $\varepsilon \approx 0.5$. Note, the location in ε of this minimum remains roughly unchanged as the inter-nodal distance h decreases (corresponding to R decreasing from 1 to 0.3 and 0.1). This invariance of the optimal ε value under refinement was noted previously in [7] for RBF-FD in the absence of supporting polynomials.

2. The middle row of subplots differs from the top one in that it was calculated using double precision (16 digit) instead of extended precision (80 digit). For these subplots, the loss of accuracy for small ε, below the dashed horizontal line indicated by the marker (||||||), corresponds closely to where the bottom row of subplots shows the condition number of the coefficient matrix in (8) to exceed about 10^{15}.

3. In double precision, using a stable algorithm such as RBF-QR [18, 25] or RBF-GA [20] gives full numerical access to small ε-values. In this context, the use of such an algorithm would be analogous to the top row of the figure, showing that the error is lowest when using small epsilon values, with adding polynomial terms having no effect (notice the -15.5 contour in

the right most subplot of the top row). Thus, there is no reason to include extra polynomials except possibly a constant, which can result in a less oscillatory interpolant.

4. In double precision, if using GA plus polynomials without employing a stable algorithm, the last two rows of subplots show that the best strategy is to lower ε until the condition number is close to 10^{15}. Including only low degree polynomials has little effect, but including close to the maximal permissible degree of polynomials increases the accuracy significantly, as shown for $R = 0.3$ and $R = 0.1$ in the middle row of subplots.

5. When including close to the maximal number of polynomials, the system in (8) has nearly doubled in size, increasing the cost of the weight calculation by around a factor of 8. For comparison, the RBF-GA algorithm increases the cost by about a factor of 10. Thus, these two approaches for reaching high accuracy roughly break even in computational cost. When calculating in the small ε regime with a stable algorithm, it is not surprising that added polynomials offer no benefit, since the GA themselves virtually span the same function space and exactly the same space in the $\varepsilon \to 0$ limit [8, 21, 28].

6. Stagnation errors arise when not using either a stable algorithm or appending with high-order polynomials. Maintaining an acceptable condition number while h is decreased requires ε to be increased, eliminating convergence.

5.2 RBF-FD using PHS with polynomials

This section describes the test results shown in Figure 7. Instead of GA RBFs, we now use the PHS RBFs $\phi(r) = r^m$ with $m = 1, 3, 5, \ldots, 21$. Tests with $\phi(r) = r^m \log r$ for $m = 2, 4, 6, \ldots$ showed no significant differences, and could have been used equally well here. Therefore, while some theory suggests using $\phi(r) = r^{\,m}$, $m = 1, 3, 5, \ldots$ in odd dimensions, and $\phi(r) = r^{\,m} \log r$, $m = 2, 4, 6, \ldots$ in even dimensions, that distinction may not be necessary.

Our observations from Figure 7 are the following:

1. The top two rows of subplots corresponding to quadruple vs. double precision, respectively, are virtually indistinguishable from each other, showing that standard double precision is completely adequate. This is in spite of the last row showing the condition number for the matrix in (8) reaching high values in different parts of the parameter domain.

2. In contrast to the GA case, the high condition number encountered here is harmless in actual computing. For PHS, it only reflects a limitation in the definition of the condition number, being sensitive to scaling issues for the rows and columns of a matrix even when these have no adverse effect on the computed interpolant. This was observed earlier in [23].

3. Increasing the polynomial degree is seen to always be beneficial for improving accuracy, whereas increasing the RBF degree matters little as the node sets are refined from $R = 1$ to $R = 0.1$. Heuristically, smooth functions under refinement appear increasingly as low-degree polynomials. As a result, the polynomial part of the RBF-FD approximations then 'takes over' (i.e. the polynomial coefficients become more dominant) from the non-smooth RBF part of the expansion. However, having RBFs present (i.e. more nodes than polynomial terms) makes singularities extremely unlikely.

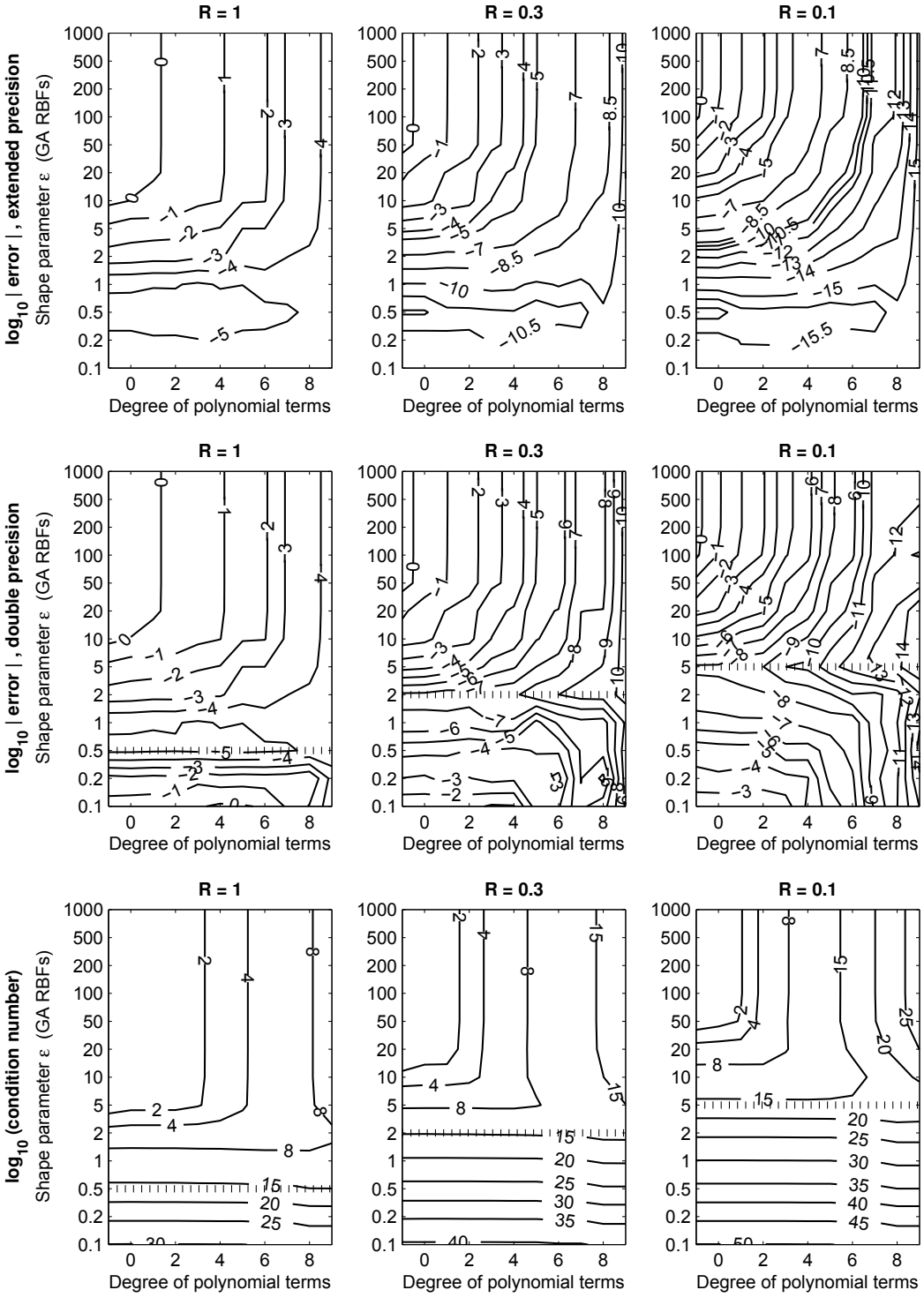

Figure 6: Top two rows: Accuracy ($\log_{10}$ error) in GA-based RBF-FD approximations, as functions of shape parameter ε and degree of included polynomial terms, calculated using 80 decimal digits extended precision vs. in standard 16 digit double precision, Bottom row: $\log_{10}$ of the condition number for the system (8) (calculated using 80 decimal digits extended precision). The level curves utilize interpolation since the polynomial degree can only take on integer values.

4. The right edges of the subplots correspond to including 2-D polynomials of degree 9. As in the GA case, nonsingularity, and accuracy with it, collapse if this is increased one more degree since there are 55 linearly independent 2-D polynomials of degree 9 and 66 of degree 10. With an $n = 56$ node stencil, the last 66 rows in the degree 10 counterpart to (8) will become linearly dependent, making the matrix singular.

5. Inspecting the left edge in the three subplots in the top or middle row, there is little, if any, error improvement as R (i.e., h) is decreased. This is a typical example of stagnation error for PHS, that is a failure of convergence under refinement. The mechanism is very different than that for GA and will be discussed in Section 5.5. Including polynomials of increasing order causes stagnation error to disappear for interpolation, then in turn for first derivatives, then second derivatives, etc.

5.3 Level of accuracy under refinement: RBF versus polynomial interpolation coefficients

Using the 56 node stencil shown in Figure 4(a) for increasingly smaller values of R, we interpolate (12) using RBFs and polynomials up to degree 4. The coefficients are separated according to the type of function they correspond to. The RBF coefficients are $\lambda_1, \lambda_2, \ldots, \lambda_{56}$, the degree 0 coefficient is μ_1, the degree 1 coefficients are μ_2, μ_3, the degree 2 coefficients are μ_4, μ_5, μ_6, and so on. In other words,

$$\begin{aligned} s(x,y) =& \lambda_1 \phi\left(\|(x-x_1, y-y_1)\|\right) + \cdots + \lambda_{56} \phi\left(\|(x-x_n, y-y_n)\|\right) \\ &+ \mu_1 \\ &+ \mu_2 x + \mu_3 y \\ &+ \mu_4 x^2 + \mu_5 xy + \mu_6 y^2 \\ &+ \cdots \end{aligned} \tag{13}$$

Our aim is to compare the relative importance of the two different parts of the interpolant (i.e the RBF coefficients versus the polynomial coefficients) as the value of R decreases. To this end, the ℓ_2 norm of each vector of coefficients is calculated for decreasing R in Figure 8. For example, the ℓ_2 norm of the RBF coefficients is $\|\lambda\|$, where $\lambda = \{\lambda_j\}_{j=1}^{56}$, for the constant term ('deg 0' in figure) simply μ_1, for the linear terms ('deg 1' in figure) $\|\mu\|$, where $\mu = \{\mu_j\}_{j=2}^{3}$, for the quadratic terms ('deg 2' in figure) $\|\mu\|$, where $\mu = \{\mu_j\}_{j=4}^{6}$, and so on.

Keeping the vertical scale fixed while refining the horizontal scale (i.e. decreasing R), the coefficients of the polynomial terms will decrease as their corresponding degree. For example the constant term will remain fixed under refinement since eventually the function will locally approach a constant. In turn, the linear terms represent the slope of the function which will decrease at the same rate R decreases, i.e. at the same rate at which the horizontal scale is being shrunk. Likewise quadratic terms will decay at the rate the curvature of the function is decaying under refinement, i.e quadratically. Thus, each polynomial term in the RBF-FD approximation will decay at the rate of its degree. Whatever is 'leftover' in the approximation is the rate of decay of the RBF coefficients, which will be $O(R^{\ell+1})$ (the order of the remainder from classical polynomial interpolation), where ℓ is the highest degree of polynomial used in the approximation. Thus under refinement, the RBF portion of the expansion plays the least significant role in the accuracy of the approximation and what degree PHS is used matters little. These observations are confirmed in Figure 8, where the

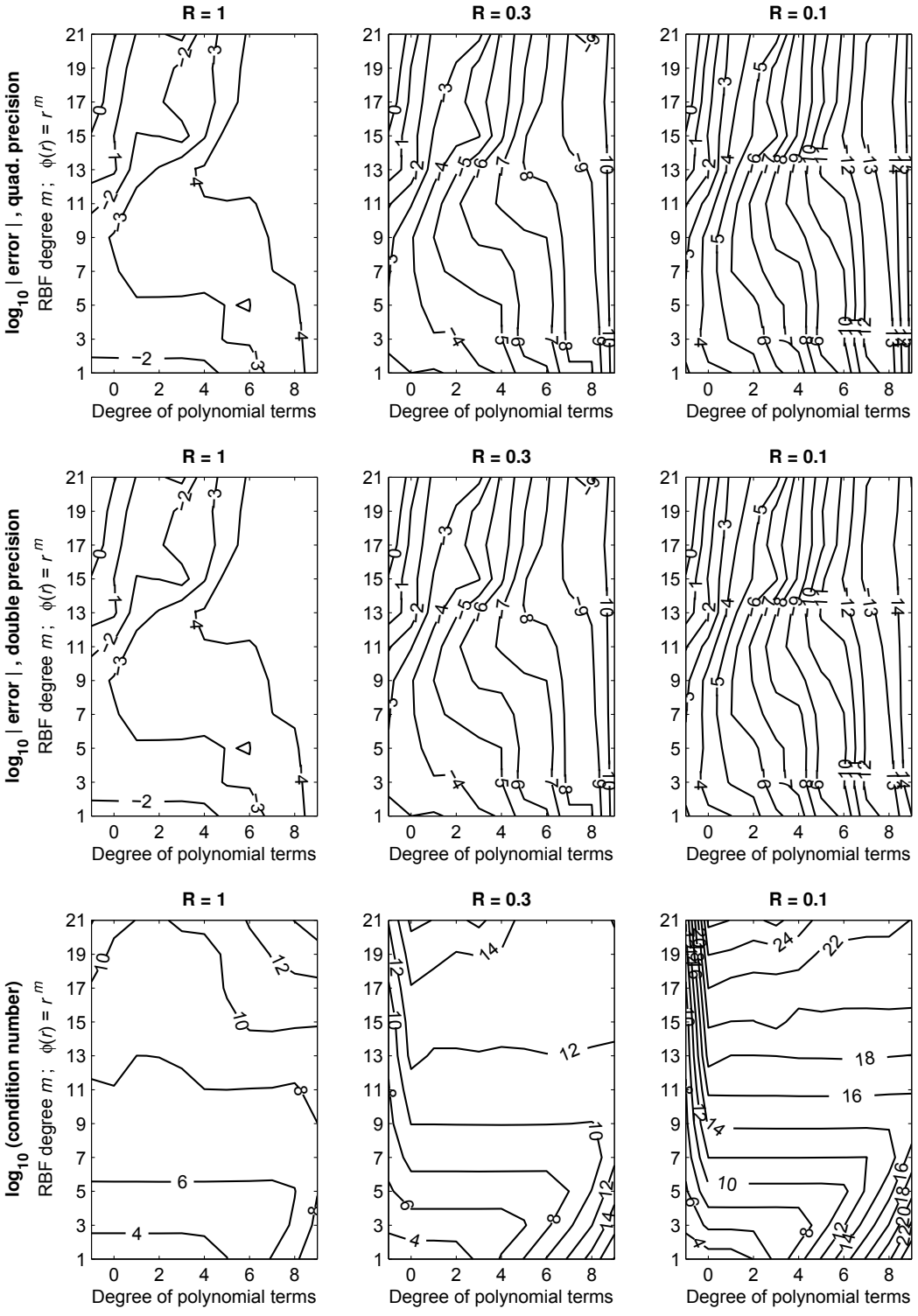

Figure 7: Top two rows: Accuracy ($\log_{10}$ error)in PHS-based RBF-FD approximations, as functions of the RBF degree and the degree of included polynomial terms, calculated using 34 decimal digits quad precision vs. in standard 16 digit double precision, Bottom row: $\log_{10}$ of the condition number for the system (8) (calculated using quad precision). As in Fig. 5, plotting the level curves utilizes interpolation as 1) the polynomial degree can only take on integer values and 2) the RBF degrees m are odd integers.

interpolation error is also plotted. Notice that adjusting the exponent of the PHS RBF, only slightly shifts the error curve ($---$) in the subplots, but does not change its slope - c.f. its position relative to the dotted $O(R^5)$ comparison curve, placed at the same in all the subplots.

While the error is slightly different in each subplot, the results confirm that adjusting the exponent of the PHS RBF might shift the error curve up or down, but does not change its slope.

Figure 9 shows that the same conclusion can be drawn for GA RBFs but with a slight twist. In this case, the shape parameter ε plays a role in determining the relative importance between the RBF and polynomial portion of the expansion. It is known that as GA RBFs go flat, i.e ε becomes small, they begin to span the same space as polynomials [21]. Hence, small values of ε yield a combined interpolant that is still heavily influenced by the RBFs at high resolution, as can been seen in Figure 9 for the $\varepsilon = 1/2$ case. However, regardless of the value of ε, the RBF coefficients always converge at the same rate as the interpolation error, just as in the PHS case.

Section 5.1

5.4 Some concluding observations of GA vs. PHS when used with polynomials

In the present test case, three choices appear quite similar both in terms of the accuracy that can be reached and cost of generating the RBF-FD stencils:

i GA without additional of polynomials, implemented for small ε by a stable algorithm [20, 25], such as RBF-GA (available at MATLAB central - titled "RBF-GA Differentiation Weights"),

ii GA with polynomials close to the maximal permissible degree, using ε values that give condition numbers around 10^{15},

iii PHS with polynomials close to the maximal permissible degree.

Using a smooth RBF type (such as GA) for low ε together with weight calculations in extended precision would also reach the same high accuracy levels, but the cost will be far higher than with the choices above and thus not realistic for practical modeling.

Not one of the above three options, (i) - (iii), will encounter stagnation errors under node refinement. Option (iii) is the easiest to implement and code and thus will be the focus of our following discussion as well as demonstration example.

5.5 Stagnation errors when using PHS with polynomials

We have seen above why GA without polynomials encounter stagnation errors under node refinement if ε is increased to keep the condition number constant (which is necessary for stability in the absence of a stable algorithm). The mechanism for PHS is completely different in that any change in the internodal distance h scales out of the interpolation problem. For example, assume we refine by a factor h. The interpolation matrix in (2) is scaled by h^{2m-1}. Since the right hand side of that system does not change implies that the coefficients λ are scaled by $h^{-(2m-1)}$. Given that the evaluation point is also scaled by h leaves the interpolant in (1) unchanged under refinement.

However, the above PHS argument only describes the mechanism by which stagnation errors occur under refinement and not the character or distribution of the error. For PHS without polynomials,

stagnation errors have a completely different character and disrupt convergence by penetrating in from the RBF-FD stencil's outer edge. The phenomenon is closely linked to spline theory. In order to gain an intuitive insight, we will consider the 1-D interpolation problem for a constant when using $\phi(r) = r^3$, r^5, r^7, etc.

To begin with, let us consider the case of $\phi(r) = r^3$, which was originally analyzed in [14]. With nodes $-1 = x_1 < x_2 < \ldots < x_n = 1$, the interpolant $s(x) = \sum_{i=1}^{n} \lambda_i |x - x_i|^3$ becomes a cubic spline with the non-intuitive end conditions (See Section A.1.2 in [16] for derivation or [14])

$$\begin{array}{rl} s''(-1) & = s'(1) - 2s'(-1) - \frac{3}{2}(s(1) + s(-1)) \\ s''(1) & = 2s'(1) - s'(-1) - \frac{3}{2}(s(1) + s(-1)) \end{array} , \tag{14}$$

which should be contrasted to the end conditions

$$s''(-1) = s''(1) = 0 \tag{15}$$

for the *natural* cubic spline. As a result, constant data $f_i = 1$ will produce a non-constant interpolant with severe errors near the edge of the domain. This is shown in Figure 10(a) together with its counterpart for $\phi(r) = r^7$. These large edge errors oscillate and decay as $O(\alpha^{|k|})$, when k steps(nodes) are taken from the stencil edge towards its interior. The geometric decay rates for these oscillations follow directly from considering the associated B-splines as is described in Appendix A, where a one line MATALB code is given to calculate the oscillation decay rate α. These are listed below for splines of odd degree.

spline degree	3	5	7	9	$\cdots$	∞
α	0.2679	0.4306	0.5353	0.6080	$\cdots$	1.0000

. (16)

Figure 10(b) shows that the amplitudes of the observed oscillations indeed decay at these predicted rates. Increasing the RBF degree is seen to offer only minor boundary improvement, which is then offset by a slower decay of the error towards the interval center. In view of this observation, it is not surprising that, in Figure 7, changing the RBF degree in the absence of polynomials had little effect on the accuracy. The interpolation error near the stencil center is simply the edge error reduced by a certain factor. Therefore, shrinking the distance h between the stencil node points will make no difference with regard to the interpolation error anywhere on the stencil and in particular not near its center, the very definition of stagnation error.

When supplementing the $\phi(r) = |r|^3$ with constant and linear terms and imposing the matching constraints $\sum_{i=1}^{n} \lambda_k = 0$ and $\sum_{i=1}^{n} \lambda_k x_k = 0$, the resulting end conditions will change from (14) to (15), i.e. the approximation becomes exact for all constant and linear data. The pattern continues: when including higher-order polynomial terms, these dominate and provide the corresponding levels of convergence under RBF-FD refinement (e.g. including up to fifth-order polynomials for a smooth function will result in sixth-order convergence, regardless of the PHS used).

5.6 Role of polynomials with PHS when approximating derivatives

This section is concerned with errors that arise when approximating derivatives, assuming that all the function values are provided. If the task at hand instead is to apply the RBF-FD approximations to solve PDEs, both numerical and time stability issues arise with regard to the number of polynomials that can be appended. These will be discussed in subsequent studies. For time-dependent problems, these include the handling of boundaries, and ensuring stability during time

stepping. For elliptic PDEs, local RBF-FD stencils will need to be combined into global matrices, raising issues such as ensuring non-singularity and avoiding spurious solutions. Some present conclusions (such as recommending the use of supplementary polynomials of nearly the maximal possible degree) will then need to be reconsidered.

Independent of the dimension, from the argument above, we would expect errors under refinement (with h being the internodal distance) to be of the form $O(h^{\ell-k+1})$ when approximating a k^{th} derivative, using $\phi(r) = r^p$, p odd, supplemented with polynomials of degree ℓ. The two parts of Figure 11 show how the error when approximating the Laplacian varies with h when using $\phi(r) = r^3$ and $\phi(r) = r^7$, respectively, in both cases together with polynomials of different degrees. We consider again the test function (12). In this case, 1000 different Halton nodes are distributed in the unit circle. A 56 node stencil with internal node spacing of approximately h is then centered at each of the Halton nodes. Of all these 1000 stencils, the worst error encountered was recorded for each h and plotted in Figure 11. The log-log plots are in excellent agreement with all the observations above - in particular with the errors being of size $O(h^{\ell-k+1})$, here $k = 2$. The thick dashed lines mark $Error = 10^{-15}/h^2$, which is the accuracy barrier in double precision that arises if weights with magnitudes adding up to $10/h^2$ are combined with function values that are uncertain to 10^{-16}. For example, the weights for the standard second-order FD approximation

$$\triangle = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \approx \begin{bmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{bmatrix} /h^2$$

add up in magnitude to $8/h^2$, with higher multiples of $1/h^2$ inevitable for more accurate approximations. The curves in Figure 11 reach right down to this barrier, showing that no significant digits have been lost due to any RBF-FD related conditioning issues. This 'barrier' appears in just the same position for regular lattice-based FD approximations.

Moreover, Figure 12 repeats the test case in Figure 11, using different stencils sizes of $n = 45$, 66 and 91. Notice that the convergence rate is $O\left(h^{l-k+1}\right)$ in all subplots, independent of the stencil size n when higher-order polynomials used. Only for low-degree polynomials do we see clear accuracy gain with increasing stencil size. This last observation is reinforced in the next section.

6 RBF-FD versus Polynomial Least Squares

Since the convergence rate appears to depend only on the degree of the polynomials used and not on the PHS, it is natural to wonder if one might achieve the same accuracy by finding weights based solely on a polynomial least-squares approximation, a method commonly used in the scientific community. In order to test this comparison, we use the same function as given in (12) over decreasing R (thus zooming into the function while keeping the vertical axis fixed). In this case, three different stencil sizes, $n = 10$, 28, and 55 (see Figure 13), are used with the approximation again being evaluated at many locations near the stencil center and the relative ℓ_2 error measured. For the comparison, polynomials up to second degree are used in both cases with PHS r^5 used for the RBF-FD approximation.

Figure 14 gives the results. The observations to note when including up to second degree polynomials are:

1. For all approximations, the rate of convergence is third-order regardless of the stencil size, i.e. we have confirmed again it is controlled by the highest degree polynomial used.

2. For polynomial least-squares approximations, increasing stencil size *does not* increase accuracy. Note that three least-squared curves are indistinguishable in Figure 14.

3. For PHS RBF-FD with polynomials approximations, increasing stencil size does increase accuracy. This is consistent with the low-degree polynomial observations in Figure 12. However, this trend becomes less clear for high-order polynomials (e.g. up to sixth degree) and very large stencils such as $n = 75$.

4. If $p = \binom{l+d}{l}$ polynomial terms are used and n is the stencil size, then polynomial least-squares systems require $O(p^2 n)$ operations to solve. Then, a consequence of point 2 is that to increase accuracy for least-squares approximations p must increase for a fixed n. Thus, as p approaches n the computational cost roughly breaks even RBF-FD systems with polynomial augmentation, which require $O(n^3)$ operations.

Even for higher-degree polynomials, the RBF-FD version continues to outperform least-squares approximations in terms of accuracy. However, the increases in accuracy with stencil size for RBF-FD become less distinct as the degree of polynomials included increases.

7 Conclusions

One thrust of this paper is to understand, in the context of RBF-FD interpolation, how polynomials can help defeat the stagnation (saturation) error. The mechanisms by which stagnation error occurs under node refinement and how it can be characterized for GA versus PHS RBF-FD approximations has been shown to be fundamentally different. However, the remedy to achieve *the highest-order convergence rate* possible for a given stencil size is similar. Below we summarize the mechanism, character, and remedy for stagnation errors for GA and PHS interpolants on RBF-FD stencils.

For GA

1. **Mechanism**: Maintaining an acceptable condition number for accuracy while the internodal distance h is decreased requires the shape parameter ε to be proportionately increased, eliminating convergence.
2. **Character**: The stagnation error oscillates uniformly throughout the domain with a period proportional to the node spacing h. The exception is at the edges of the stencil where Runge phenomena pollutes the interpolant, causing large errors.
3. **Remedy**: There are two options: 1) Use a stable algorithm as RBF-GA or 2) Using ε values that give condition numbers for the interpolation matrix of about 10^{15}, add all polynomials up to around the maximal degree permissible for the stencil size.

For PHS

1. **Mechanism**: Any refinement of h simply scales out of the interpolation problem since the interpolation matrix gets scaled by h^{2m-1}, implying that the expansion coefficients λ are scaled by $h^{-(2m-1)}$. Since the evaluation point is also scaled by h, the interpolant remains unchanged.
2. **Character**: When not including polynomials, large oscillating edge errors penetrate the stencil's interior at a given decay rate known from spline theory. Thus, the interpolation error source at the stencil's center is simply the edge error reduced by a given factor and independent of internodal distance.

3. **Remedy**: Include all polynomials up to about the maximal degree permissible for the stencil size.

It is important to note that for interpolation purposes, adding polynomials of any degree will defeat stagnation error but at a convergence rate given by the highest degree polynomial added. Why adding polynomials work in combatting stagnation error is also detailed in the paper and can be summarized as follows:

> Under node refinement (i.e. zooming in locally), the function will behave more as a polynomial, eventually approaching a constant. Thus each polynomial term in the RBF-FD approximation will decay according to its degree, with the constant term never decaying. Whatever is 'leftover' in the approximation is the decay rate of the RBF coefficients and will also be the rate of decay for the interpolation error. That is, the decay rate of the RBF coefficients will be the same order as that of the remainder term from classical polynomial interpolation.

Given the observations above, simply adding polynomials to PHS, and not having to consider a shape parameter or condition number, makes them very attractive to work with. Hence, the rest of the paper considered the implications on RBF-FD approximations when using such a PHS-polynomial basis and can be summarized as follows:

- There is major systematic improvement of accuracy by including polynomials of increasing degree for both approximating the interpolant and its derivatives.
- Since the polynomials control the rate of convergence, increasing the degree of the PHS gives little, if any benefit.
- Under refinement, the convergence rate is independent of stencil size.
- When augmenting PHS with high-order polynomials, accuracy is not improved with increasing stencil size; however with low-order polynomial augmentation stencil size does improve accuracy but not convergence rate.
- In terms of accuracy, PHS RBF-FD systematically outperforms polynomial least-squares approximations for the same stencil size.

Again, it is to be emphasized that this paper is focused on interpolation. It will be shown in follow-up studies that when using polynomials in RBF-FD approximations for solving elliptic and hyperbolic PDEs, it is not always advantageous to use very high-order polynomials, i.e. those close to the degree permissible by the stencil size. In fact, numerical stability for elliptic PDEs and time stability for hyperbolic PDEs seem to be more robust when the number of polynomials is roughly half the stencil size.

Acknowledgments:

The presented research was supported by the NSF grants DMS-0934317, OCI-0904599 and by Shell International Exploration and Production, Inc.

Appendix A: A note on Cardinal Spline decay rates

A cardinal spline on an infinite equispaced grid takes the value one at one node and zero at all other nodes. Like any other spline, it can be written as a linear combination of translates of the corresponding B-spline. Away from the central node, these B-splines must add up to zero, giving a linear recursion relation between their successive coefficients. The leading-order decay rate of the amplitudes of the in-between node point oscillations will be the same as the decay rate of these B-spline coefficients. As an example, the cubic B-spline takes at its interior nodes values proportional to 1, 4, 1. The corresponding characteristic polynomial $p(x) = 1 + 4x + x^2$ has the two roots $x_{1,2} = -2 \pm \sqrt{3}$. Both roots are real and negative. We obtain the oscillation decay rate from the root with the largest magnitude that still obeys $|x_i| < 1$, that is $O((-2 + \sqrt{3})^k) \approx O((0.28795)^k)$ when k nodes away from the center node. This observation generalizes immediately to splines of any degree m. Implemented in MATLAB, we obtain the decay rate by executing the statement

```
roots(ppval(bspline(0:m+1),1:m))'
```

For example, this produces for $m = 7$ the output

```
ans =
  -109.3052   -8.1596   -1.8682   -0.5353   -0.1226   -0.0091
```

from which we read off the asymptotic oscillation decay rate for splines of degree 7 as $O((-0.5353)^k)$.

References

[1] V. Bayona, N. Flyer, G. M. Lucas, and A. J. G. Baumgaertner, *A 3-D RBF-FD solver for modeling the atmospheric global electric circuit with topography (GEC-RBFFD v1.0)*, Geosci. Model Dev **8** (2015), 3007–3020.

[2] V. Bayona and M. Kindelan, *Propagation of premixed laminar flames in 3D narrow open ducts using RBF-generated finite differences*, Combustion Theory and Modelling **17** (2013), 789–803.

[3] E. Bollig, N. Flyer, and G. Erlebacher, *Solution to PDEs using radial basis function finite-differences (RBF-FD) on multiple GPUs*, J. Comput. Phys. **231** (2012), 7133–7151.

[4] M. D. Buhmann, *Radial basis functions: Theory and implementations*, 12. Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, Cambridge, 2003.

[5] G. Chandhini and Y.V.S.S. Sanyasiraju, *Local RBF-FD solutions for steady convection-diffusion problems*, Int. J. Num. Meth. Eng. **72** (2007), 352–378.

[6] P. P. Chinchapatnam, K. Djidjeli, P. B. Nair, and M. Tan, *A compact RBF-FD based meshless method for the incompressible Navier-Stokes equations*, J. Eng. Maritime Env. **223** (2009), 275–290.

[7] O. Davydov and D. T. Oanh, *Adaptive meshless centres and RBF stencils for Poisson equation*, J. Comput. Phys. **230** (2011), 287–304.

[8] T. A. Driscoll and B. Fornberg, *Interpolation in the limit of increasingly flat radial basis functions*, Comput. Math. Appl. **43** (2002), 413–422.

[9] G. Erlebacher, E. Saule, N. Flyer, and E. Bollig, *Acceleration of derivative calculations with application to radial basis function: finite-differences on the intel mic architecture*, ICS '14 Proceedings of the 28th ACM international conference on Supercomputing (New York, NY), ACM, 2014, pp. 263–272.

[10] G. E. Fasshauer, *Meshfree Approximation Methods with MATLAB*, Interdisciplinary Mathematical Sciences - Vol. 6, World Scientific Publishers, Singapore, 2007.

[11] N. Flyer, *Exact polynomial reproduction for oscillatory radial basis functions on infinite lattices*, Comput. Math. Appl. **51** (2006), 1199–1208.

[12] N. Flyer, G.A. Barnett, and L.J. Wicker, *Enhancing finite differences with radial basis functions: Experiments on the Navier-Stokes equations*, J. Comput. Phys. **316** (2016), 39–62.

[13] N. Flyer, E. Lehto, S. Blaise, G. B. Wright, and A. St-Cyr, *A guide to RBF-generated finite differences for nonlinear transport: Shallow water simulations on a sphere*, J. Comput. Phys **231** (2012), 4078–4095.

[14] B. Fornberg, T. A. Driscoll, G. Wright, and R. Charles, *Observations on the behavior of radial basis functions near boundaries*, Comput. Math. Appl. **43** (2002), 473–490.

[15] B. Fornberg and N. Flyer, *Accuracy of radial basis function interpolation and derivative approximations on 1-D infinite grids*, Adv. Comput. Math. **23** (2005), 5–20.

[16] ———, *A Primer on Radial Basis Functions with Applications to the Geosciences*, SIAM, Philadelphia, 2015.

[17] ———, *Solving PDEs with radial basis functions*, Acta Numerica **24** (2015), 215 –258.

[18] B. Fornberg, E. Larsson, and N. Flyer, *Stable computations with Gaussian radial basis functions*, SIAM J. Sci. Comput. **33(2)** (2011), 869–892.

[19] B. Fornberg, E. Larsson, and G. B. Wright, *A new class of oscillatory radial basis functions*, Comput. Math. Appl. **51** (2006), 1209–1222.

[20] B. Fornberg, E. Lehto, and C. Powell, *Stable calculation of Gaussian-based RBF-FD stencils*, Comp. Math. Applic. **65** (2013), 627–637.

[21] B. Fornberg, G. Wright, and E. Larsson, *Some observations regarding interpolants in the limit of flat radial basis functions*, Comput. Math. Appl. **47** (2004), 37–55.

[22] B. Fornberg and J. Zuev, *The Runge phenomenon and spatially variable shape parameters in RBF interpolation*, Comput. Math. Appl. **54** (2007), 379–398.

[23] A. Iske, *On the approximation order and numerical stability of local Lagrange interpolation by polyharmonic splines*, Modern Developments in Multivariate Approximation (W. Haussmann, K. Jetter, M. Reimer, and J. Stöckler, eds.), International Series of Numerical Mathematics, vol. 145, Birkhäuser Verlag, Basel, 2003, pp. 153–165.

[24] S. H. Javaran, N. Khaji, and A. Noorzad, *First kind Bessel function (J-Bessel) as radial basis function for plane dynamic analysis using dual reciprocity boundary element method*, Acta Mech. **218** (2011), 247–258.

[25] E. Larsson, E. Lehto, A. Heryudono, and B. Fornberg, *Stable computation of differentiation matrices and scattered node stencils based on Gaussian radial basis functions*, SIAM J. Sci. Comput. **35** (2013), A2096–A2119.

[26] J. Nocedal and S. Wright, *Numerical optimization*, Springer Science & Business Media, 2006.

[27] C. M. C. Roque, D. Cunha, D. Shu, and A. J. M. Ferreira, *A local radial basis functions - Finite difference technique for the analysis of composite plates*, Eng. Anal. Bound. Elem. **35** (2011), 363–374.

[28] R. Schaback, *Multivariate interpolation by polynomials and radial basis functions*, Constr. Approx. **21** (2005), 293–317.

[29] Y. Y. Shan, C. Shu, and Z. L. Lu, *Application of local MQ-DQ method to solve 3D incompressible viscous flows with curved boundary*, Comp. Mod. Eng. & Sci. **25** (2008), 99–113.

[30] V. Shankar, G. B. Wright, R. M. Kirby, and A. L. Fogelson, *A radial basis function (RBF)-finite difference (FD) method for diffusion and reaction-diffusion equations on surfaces*, J. Sci. Compu. **63(3)** (2015), 745–768.

[31] Y. Shi and C. H. Chan, *Improved 3D full-wave Green's function interpolation method*, Elec. Lett. **47(3)** (2011), 174–175.

[32] D. Stevens, H. Power, M. Lees, and H. Morvan, *The use of PDE centers in the local RBF Hermitean method for 3D convective-diffusion problems*, J. Comput. Phys. **228** (2009), 4606–4624.

[33] M. Tillenius, E. Larsson, E. Lehto, and N. Flyer, *A scalable RBF-FD method for atmospheric flow*, J. Comput. Phys. **288** (2015), 406–422.

[34] Z. H. Wang, Z. Huang, W. Zhang, and G. Xi, *A meshless local radial basis function method for two-dimensional incompressible Navier-Stokes equations*, Numer. Heat Trans. Part B **67** (2015), 320–337.

[35] H. Wendland, *Scattered Data Approximation*, Cambridge Monographs on Applied and Computational Mathematics, vol. 17, Cambridge University Press, Cambridge, 2005.

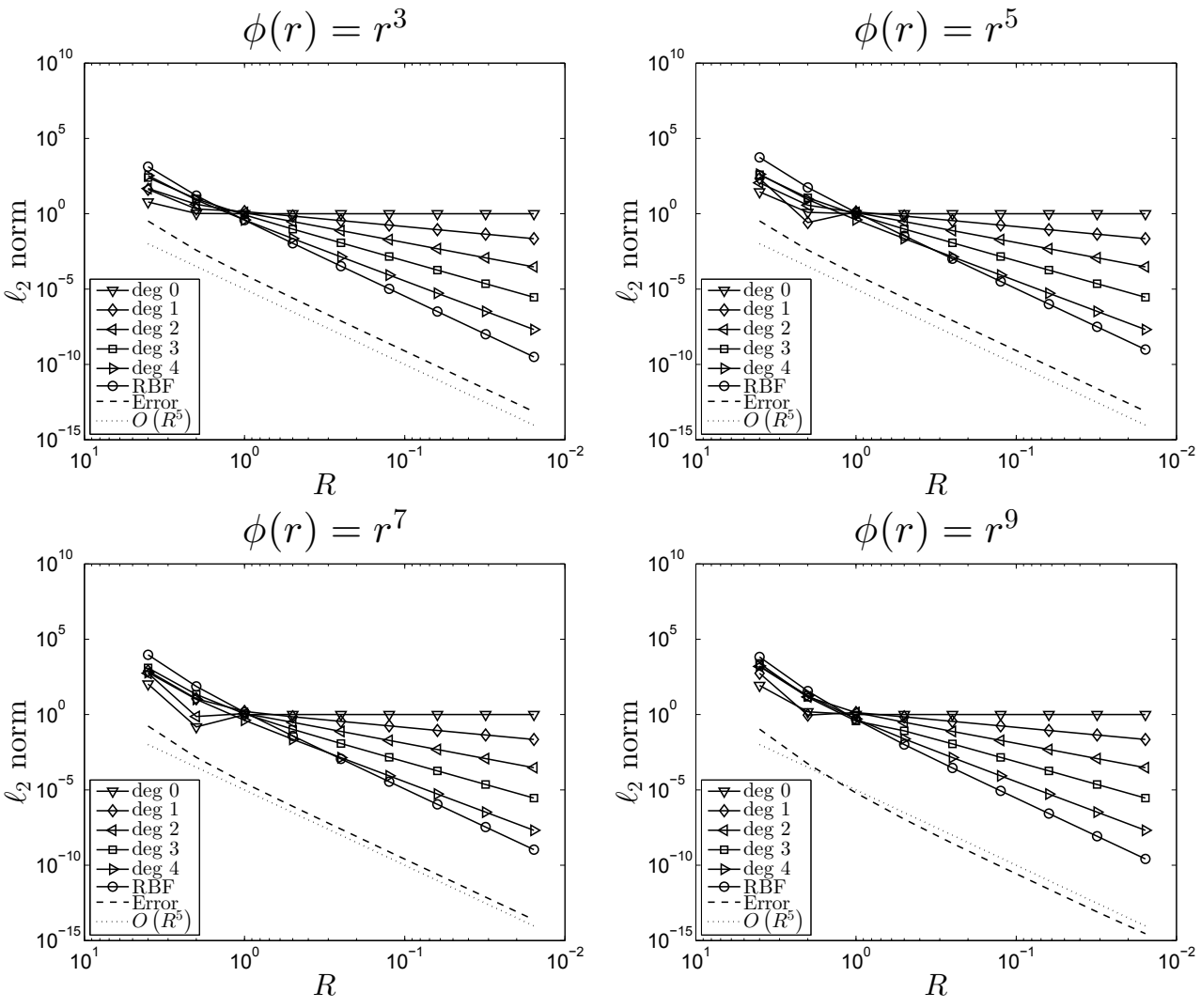

Figure 8: Behavior of the PHS and polynomial coefficients under refinement, where the latter have been separated according to polynomial degree. For the RBFs r^3, r^5, r^7, and r^9, the condition numbers of the collocation matrix are 3×10^4, 5×10^5, 5×10^6, and 3×10^7, respectively. In every case, the RBF coefficients converge to zero at the same rate as the interpolation error.

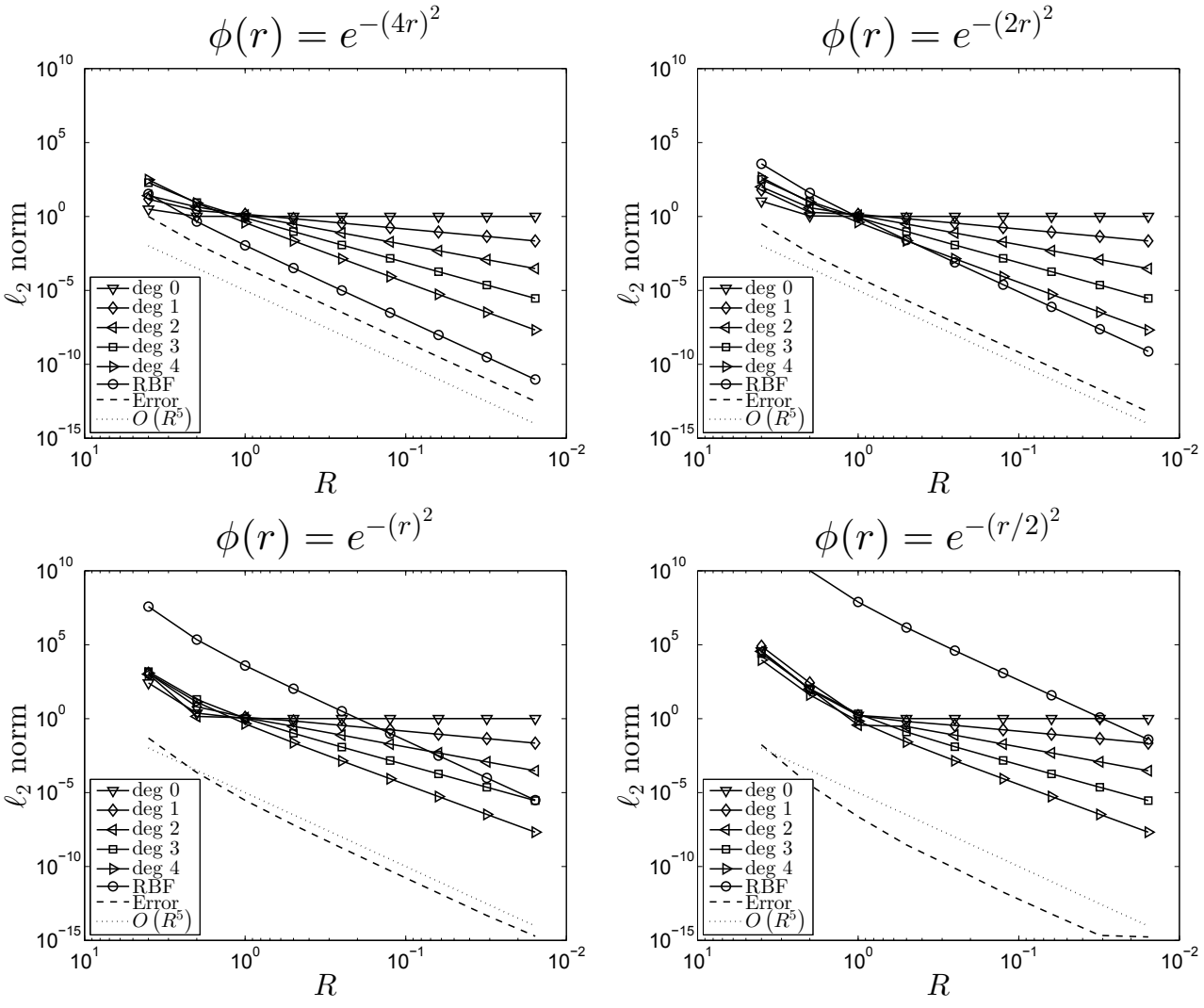

Figure 9: Behavior of the GA and polynomial coefficients under refinement, where the latter have been separated according to polynomial degree. For the shape parameters 4, 2, 1, and 1/2, the condition numbers of the interpolation matrix are 7×10^2, 9×10^4, 2×10^{10}, and 1×10^{16}, respectively. In every case, the RBF coefficients converge to zero at the same rate as the interpolation error.

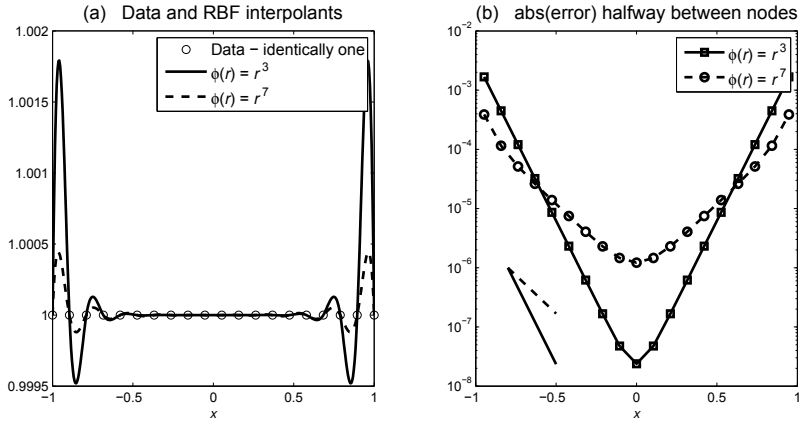

Figure 10: (a) The RBF interpolants for constant equi-spaced data using $n = 20$ nodes and PHS $\phi(r) = r^3$ and $\phi(r) = r^7$. (b) The errors at the half-way points between the nodes, corresponding roughly to the extrema of the oscillations in the interpolants. The errors decrease towards the interval center closely following the theoretical rates (16) for the two corresponding spline cases, shown by the short straight line segments at the bottom left.

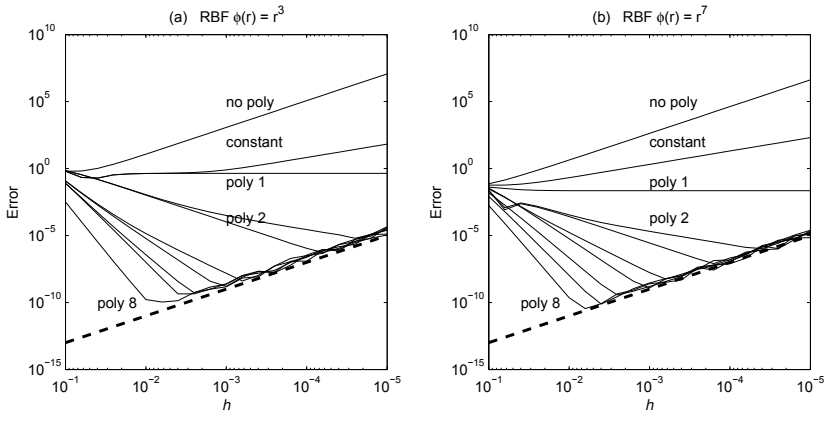

Figure 11: The worst error encountered when approximating the Laplacian of (12) using the $n = 56$ node PHS-based RBF-FD stencils, as described in the text. The curve legends ('no poly', 'constant', etc.) are in exactly the same locations in the two subplots, illustrating that the change when going from $\phi(r) = r^3$ to $\phi(r) = r^7$ indeed is minimal. The slopes of the different curves perfectly match the predictions (including the thick dashed lines, which mark $Error = 10^{-15}/h^2$).

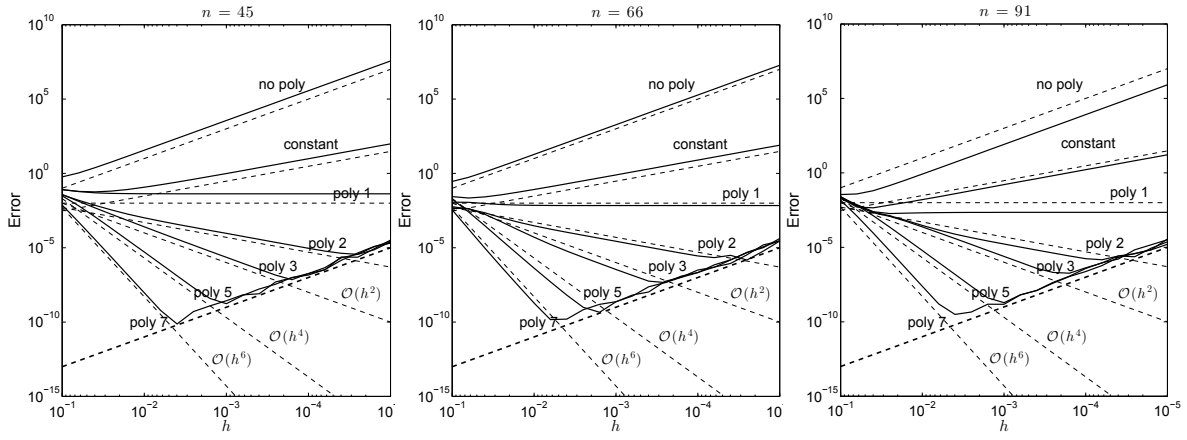

Figure 12: The worst error encountered when approximating the Laplacian of (12) using RBF-FD stencils based on r^7 + polynomials as described in the text. The thin dashed lines represent the convergence order $O\left(h^{l-k+1}\right)$ for each polynomial order l. Notice that the convergence rate is independent of the stencil size and only depends on the degree of polynomial terms included. Only for polynomial up to degree two do we see any clear accuracy gain with stencil size. As in the previous figure, the thick dashed line marks the round-off limit $10^{-15}/h^2$.

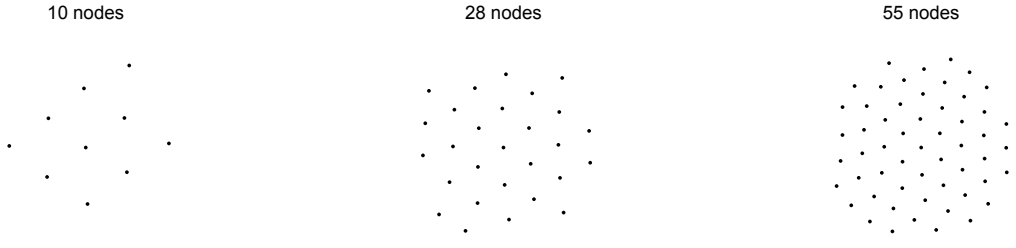

Figure 13: The three different stencils used for approximation of (12). In all cases, the error is calculated over evaluation points near the stencil center.

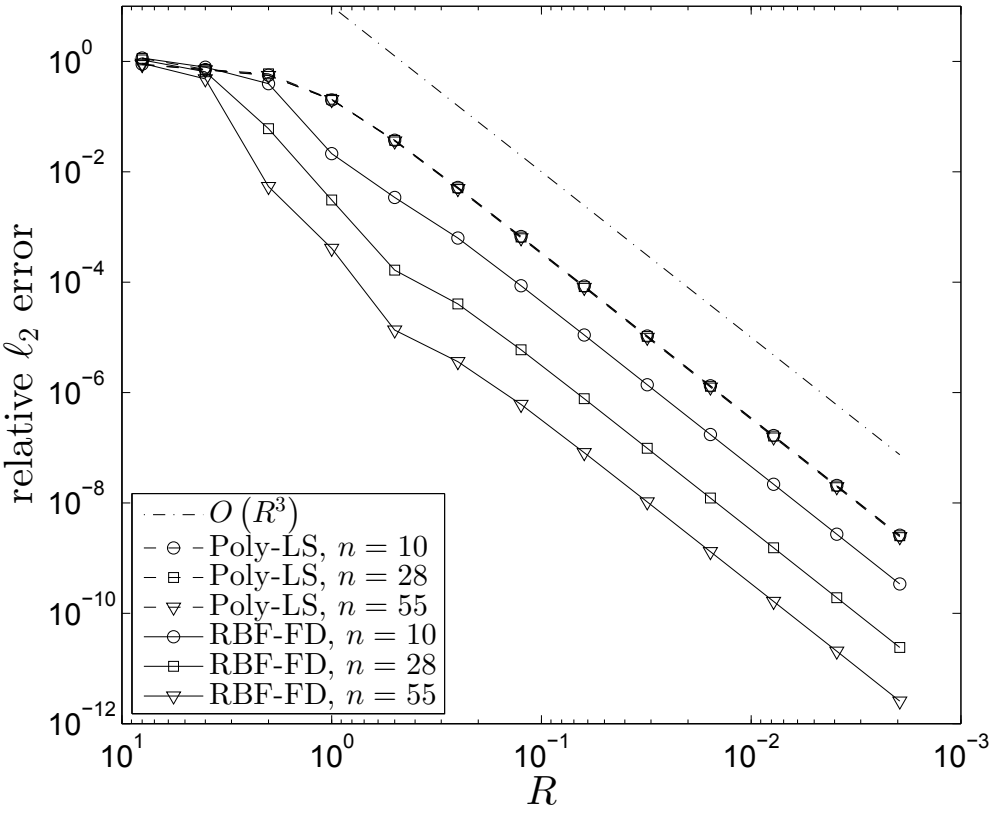

Figure 14: Convergence comparison for approximation of the test function in (12). In all cases, polynomials up to degree two are included, and in the RBF-FD cases PHS $\phi(r) = r^5$ are used. Note that increasing the stencil size has no effect on the polynomial least squares error (the three least-square error curves are indistinguishable in the figure.) Including RBFs with the polynomials does not change the rate of convergence, but greatly improves the error for each value of R as the stencil is increased, consistent with the low polynomial degree curves in Figure 12.